PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 :
G06F 17/30

A1

(11) International Publication Number: WO 98/26357

(43) International Publication Date: 18 June 1998 (18.06.98)

(21) International Application Number: PCT/CA97/00970

(22) International Filing Date: 9 December 1997 (09.12.97)

(30) Priority Data:
08/769,929 9 December 1996 (09.12.96) US

(71) Applicant *(for all designated States except US)*: PRACTICAL APPROACH CORPORATION [CA/CA]; Suite C-210, 151 Frobisher Drive, Waterloo, Ontario N2V 2C9 (CA).

(72) Inventor; and
(75) Inventor/Applicant *(for US only)*: REDFERN, Darren, M. [CA/CA]; 230 Brunswick Street, Stratford, Ontario N5A 3M4 (CA).

(74) Agents: STRATTON, Robert, P. et al.; Gowling, Strathy & Henderson, Suite 4900, Commerce Court West, Toronto, Ontario M5L 1J3 (CA).

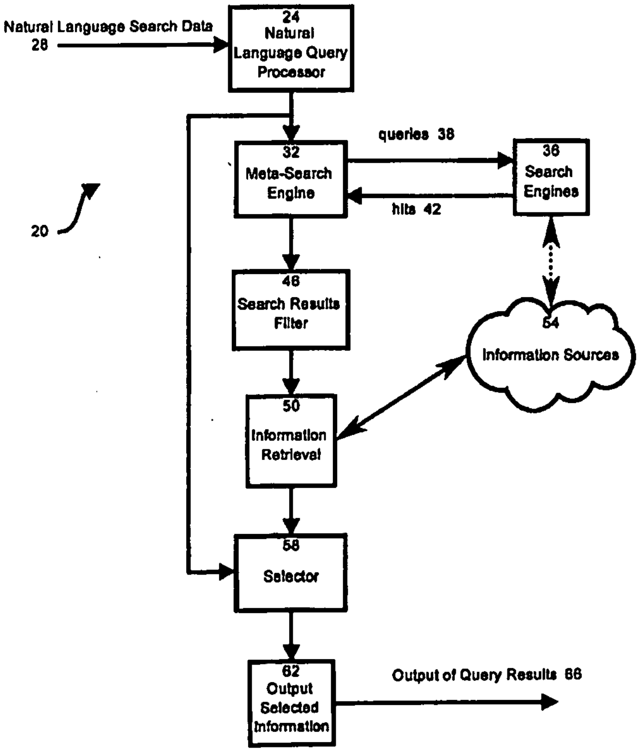
(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, HU, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG).

Published
With international search report.

(54) Title: NATURAL LANGUAGE META-SEARCH SYSTEM AND METHOD

(57) Abstract

A meta-search system accepts natural language queries which are parsed to extract relevant content, this relevant content being formed into queries suitable for each of a selected number of search engines and being transmitted thereto. The results from the search engines are received and examined and a selected number of the information sources represented therein are obtained. These obtained information sources are then examined to rank their relevance to the extracted relevant content and the portions of interest in each of these ranked information sources are determined. The determined portions are output to the user in ranked order, having first been processed to clean up the portions to include valid formatting and complete paragraphs and/or sentences.



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania
AM	Armenia
AT	Austria
AU	Australia
AZ	Azerbaijan
BA	Bosnia and Herzegovina
BB	Barbados
BE	Belgium
BF	Burkina Faso
BG	Bulgaria
BJ	Benin
BR	Brazil
BY	Belarus
CA	Canada
CF	Central African Republic
CG	Congo
CH	Switzerland
CI	Côte d'Ivoire
CM	Cameroon
CN	China
CU	Cuba
CZ	Czech Republic
DE	Germany
DK	Denmark
EE	Estonia
ES	Spain
FI	Finland
FR	France
GA	Gabon
GB	United Kingdom
GE	Georgia
GH	Ghana
GN	Guinea
GR	Greece
HU	Hungary
IE	Ireland
IL	Israel
IS	Iceland
IT	Italy
JP	Japan
KE	Kenya
KG	Kyrgyzstan
KP	Democratic People's Republic of Korea
KR	Republic of Korea
KZ	Kazakstan
LC	Saint Lucia
LI	Liechtenstein
LK	Sri Lanka
LR	Liberia
LS	Lesotho
LT	Lithuania
LU	Luxembourg
LV	Latvia
MC	Monaco
MD	Republic of Moldova
MG	Madagascar
MK	The former Yugoslav Republic of Macedonia
ML	Mali
MN	Mongolia
MR	Mauritania
MW	Malawi
MX	Mexico
NE	Niger
NL	Netherlands
NO	Norway
NZ	New Zealand
PL	Poland
PT	Portugal
RO	Romania
RU	Russian Federation
SD	Sudan
SE	Sweden
SG	Singapore
SI	Slovenia
SK	Slovakia
SN	Senegal
SZ	Swaziland
TD	Chad
TG	Togo
TJ	Tajikistan
TM	Turkmenistan
TR	Turkey
TT	Trinidad and Tobago
UA	Ukraine
UG	Uganda
US	United States of America
UZ	Uzbekistan
VN	Viet Nam
YU	Yugoslavia
ZW	Zimbabwe

Natural Language Meta-Search System and Method

FIELD OF THE INVENTION

The present invention relates to a system and method of processing queries for information. More specifically, the present invention relates to a meta-search system and method for accepting a natural language query which is processed to retrieve information from one or more information sources via at least one search engine and to extract relevant portions of those information sources for output to the originator of the query.

BACKGROUND OF THE INVENTION

Systems and methods for locating information in databases are known. An area in which such systems and methods have recently become quite common and heavily used is in searching for information on the World Wide Web (WWW) and/or on other internet sources.

Typically, an internet user will access a search engine, such as AltaVista or Yahoo through a web page maintained for that purpose by the host of the search engine and will input search data relating to the information sought into the search engine. The search data can, for example, comprise keywords or phrases related to the information sought and boolean operators to further qualify the search. Examples of such search data are, "AZT and Toxicity", wherein AZT is one keyword, Toxicity is another and the 'and' is boolean operator requiring both keywords to be present in the information source for it to be considered a match.

Once search data is input, the search engine then consults one or more indices it maintains of web pages or other information sources that match the search data. A listing of the information sources that match the search data, often referred to as "hits", is then displayed to the user, the number of matches usually being limited to some predefined maximum number. These matches are typically ranked, usually according to the number of occurrences of keywords or phrases in the information source. Generally, the information which is displayed to the user for each match comprises a location at which the document can be accessed (a URL for a WWW document) and some minimal additional information such as a document title, etc.

Generally, such search engines provide a skilled user with reasonable results from well defined and/or homogeneous databases or other information sources. For example, the APS U.S. Patent database can be efficiently searched based on the contents of well-defined information fields, such as Patent Number, Inventor Name, etc. to locate information sought.

However, while such search engines can generally provide a skilled user with reasonable results from such well defined and/or homogeneous databases, they do suffer from disadvantages. Specifically, when searching databases or information sources which are not homogeneous or well defined, such as the WWW and/or internet, even the best formed search strategy can result in a hundred or more matches, many of which are not useful to the user but which must still be reviewed by the user, to at least some extent, to determine this. Further, such search engines generally require the user to understand and be comfortable with boolean type searches and are limited to this type of search operation.

To enhance the chances that the desired information will in fact be located, a user will often perform the same search on multiple search engines thus exacerbating the number of matches which must be reviewed by the user. The use of more than one search engine can also require the user to redraft his search data to accommodate different search data requirements and/or capabilities of the different search engines. For example, some search engines may only allow keyword-based searches while others may permit searching based upon phrases.

These difficulties often result in the less skilled user not obtaining acceptable search results without multiple and/or recursive search attempts, which has led many users to adopt the interactive search technique commonly referred to as, "surfing the web" which, while often entertaining and/or informative, can be time consuming and may still not locate the desired information.

Natural Language Query (NLQ) systems are also known and are used for a variety of purposes. Generally, a NLQ system accepts a search sentence or phrase in common everyday (natural) language and parses the input sentence or phrase in an attempt to extract meaning from it. For example, a natural language search phrase used with a company's financial database may be "Give me a list of the fourth quarter general ledger expense accounts." This sentence will be processed by the NLQ system to determine the information required by the user which is then retrieved from the financial database as necessary. However, such NLQ systems are computationally expensive to operate as the processing required to determine the meaning of a sentence or phrase is significant. Further, such systems are generally limited in terms of the scope of the information which they can access. For example, a different NLQ system is likely required to correctly process queries relating to a company's financial information than is required to search a medical database of obscure diseases. Also, such NLQ systems generally only produce acceptable results with well defined and/or homogeneous databases.

It is desired to have a meta-search engine which will accept natural language search data to search for information from one or more information sources which need not be homogeneous or well defined, the meta-search engine would identify portions of the matching information which it determines to be relevant to the search data and would display at least those determined portions to the user.

SUMMARY OF THE INVENTION

It is an object of the present invention to provide a novel meta-search system and method for obtaining information relevant to a natural language query from a plurality of information sources which obviates or mitigates at least one disadvantage of the prior art.

According to a first aspect of the present invention, there is provided a method of locating information in at least one information source, comprising the steps of:

(i) accepting a natural language query describing desired information;

(ii) parsing said natural language query to extract terms relevant to said desired information;

(iii) creating search data from said extracted terms in an form appropriate to each of at least one search engines and transferring said created search data thereto to initiate a search;

(iv) receiving results comprising at least a list of information sources from each of said at least one search engines and removing redundancies therefrom to obtain a reduced list of information sources;

(v) retrieving complete copies of each information source in said reduced list;

(vi) examining each said retrieved complete copy relative to said extracted terms to determine a match ranking therefor and to identify relevant portions of said information source; and

(vii) providing said identified relevant portions to said user in order of said determined rankings.

Preferably, at least two search engines are employed. Also preferably, the extraction of relevant terms by the natural language parser includes adding terms which are alternatives and/or synonyms to terms directly extracted from the natural language query. Also preferably, the relevant portions provided to the user are at least complete paragraphs of information.

According to another aspect of the present invention, there is provided a meta-search system comprising:

a natural language query processor to produce a set of relevant terms from a natural language query;

a meta-search engine means to communicate with said at least one search engine, to form from said relevant terms a search data set for each said at least one search engine which is in a format defined for said at least one search engine and to receive search results from said at least one search engine;

filter means to remove redundancies from said received search results to produce a reduced list of identified information sources;

information retrieval means to retrieve said identified information sources;

selection means to examine each information source retrieved by said information retrieval means and to rank each said information source relative to said set of relevant terms and to identify portions of said each said information source relevant to said extracted terms; and

output means to provide said user with said identified portions in order of said ranking.

Preferably, at least two search engines are employed. Also preferably, the extraction of relevant terms by the natural language query processor includes adding terms which are alternatives and/or synonyms to terms directly extracted from the natural language query. Also preferably, the identified portions output to the user are at least complete paragraphs of information.

BRIEF DESCRIPTION OF THE DRAWINGS

Preferred embodiments of the present invention will now be described, by way of example only, with reference to the attached Figures, wherein:

Figure 1 shows a schematic representation of a meta-search system in accordance with the present invention;

Figure 2 shows a schematic representation of a natural language query processor in accordance with an embodiment of the present invention;

Figure 3 shows a schematic representation of a classification step of the natural language query processor of Figure 2;

Figures 4a through 4e show schematic representations of a manipulation step of the natural language query processor of Figure 2;

Figures 5, 5a and 5b show schematic representations of a meta search engine in accordance with an embodiment of the present invention;

Figures 6, 6a, 6b and 6c show schematic representations of a selector in accordance with an embodiment of the present invention;

Figure 7 shows a schematic representation of an HTML clean up step in the selector of Figures 6, 6a, 6b and 6c; and

Figure 8 shows a schematic representation of a text clean up step in the selector of Figures 6, 6a, 6b and 6c.

DETAILED DESCRIPTION OF THE INVENTION

Figure 1 shows a meta-search system 20 in accordance with an embodiment of the present invention. As used herein, the term "meta-search" system and/or method is intended to comprise a search system and/or method which acts between a user and one or more search engines. As described below, the meta-search system can accept a natural language query, extract relevant terms and/or phrases from that query to produce search queries appropriate to each of one or more search engines. The meta-search system has one or more of these search engines process a search query or queries to provide the meta-search system with a list of 'hits'. The meta-search engine accumulates these hits and examines them to remove redundancies. A copy of the complete information source is retrieved for a pre-selected number of the non-redundant hits and these copies are examined by the meta-search engine to determine a ranking for each information source and to determine the portions of the information source which relate to the extracted relevant terms. These portions are output to the user, in ranked order, as the results of the search.

As shown in Figure 1, system 20 includes a Natural Language Query Processor 24 which is operable to receive Natural Language Search Data 28 and to extract relevant terms and/or phrases therefrom. Specifically, search data 28 can comprise one or more complete or incomplete sentences which processor 24 parses.

Referring to Figure 2, the parsing process 100 employed by processor 24 is shown. At step 104, search data 28 is accepted and processed to remove punctuation. At step 108, groups (words and/or phrases) are classified according to a preselected classification scheme. Next, the classified groups are manipulated at step 112 to obtain a list of extracted relevant terms and this list is expanded, at step 116, to convert groups of less common phrases into more common phrases.

Specifically, at step 104 search data 28 is examined to remove all trailing punctuation, such as "?", "!" and ".", including any of these appearing before a closing single or double quotation mark. Next, all commas, colons, semi-colons are removed and any "abandoned" punctuation, defined by spaces, returns or linefeeds on either or both sides, is removed. An example of abandoned punctuation is the hyphen in "take a break - today". Processing then proceeds to step 108 which is described below, with reference to Figure 3.

Figure 3 illustrates sub-steps of step 108 wherein, at step 200, each group between quotation marks is classified as a quote() and the corresponding quotation marks are then removed from search data 28, i.e. - "grand canyon" is classified as quote(grand canyon).

At step 204, a comparison is performed between the processed search data 28 and a list of null content phrases, referred to by the present inventor as "throw away phrases". Each match between a group in processed search data 28 (other than groups classified as quote()) and the list of null content phrases is classified as a throw(). Example lists of null content phrases and null content words, in accordance with an embodiment of the present invention, are included herewith as Tables 1 and 2 respectively in Appendix A.

Next, at step 208, an "or" expansion is performed if required. An "or" expansion is intended to convert phrases such as "big/huge/jumbo" into distinct terms separated by or's, i.e. - "big or huge or jumbo".

Next, each word in processed search data 28 which has not been classified as being part of a quote() or a throw() is examined and categorized. An example of a set of categories used in a present embodiment of the invention includes: quote(), throw(), capital(), number(), join(), prep(), adject(), qword(), or(), rank1() and phrase(), of which quote() and throw() are discussed above and the remainder of which are described below. Classification proceeds in the order given above, with classification of groups as capital()'s being considered before number()'s, etc.

At step 212 each remaining unclassified word is examined to determine if it is within the definition of the capital() category. Specifically, if the first character of the word is capitalized, the word is classified as a capital(). Adjacent words which have been classified as capital()'s are combined into groups which are then classified as capital(), i.e. - capital(Mickey) capital(Mouse) are combined to capital (Mickey Mouse).

At step 216, each remaining unclassified word is examined to determine if it is within the definition of the number() category. Specifically, if the first character of the word is a number, the word is classified as a number().

At step 220, each remaining unclassified word is examined to determine if it is within the definition of the join() category. Specifically, the word is compared to a predefined list of words and, if the word is present in the list, the word is classified as a join(). An example of a list of words which are used for classifying join()'s in accordance with an embodiment of the present invention is included herewith as Table 3 in Appendix A.

At step 224, each remaining unclassified word is examined to determine if it is within the definition of the prep() category. Specifically, the word is compared to a predefined list of words and, if the word is present in the list, the word is classified as a prep(). An example of a list of words which are used for classifying prep()'s in accordance with an embodiment of the present invention is included herewith as Table 4 in Appendix A.

At step 228, each remaining unclassified word is examined to determine if it is within the definition of the adject() category. Specifically, the word is compared to a predefined list of words and, if the word is present in the list, the word is classified as a adject(). An example of a list of words which are used for classifying adject()'s in accordance with an embodiment of the present invention is included herewith as Table 5 in Appendix A.

At step 232, each remaining unclassified word is examined to determine if it is within the definition of the qword() category. Specifically, the word is compared to a predefined list of words and, if the word is present in the list, the word is classified as a qword(). An example of a list of words which are used for classifying qword()'s in accordance with an embodiment of the present invention is included herewith as Table 6 in Appendix A.

At step 236, each remaining unclassified word is then deemed to be a phrase(). Adjacent words in processed search data 28 which are categorized as phrase()'s are combined to form phrases which are then categorized as phrase().

Finally, at step 240, the first word of each classified quote() is examined to determine if it is capitalized. If it is, it is converted to lowercase and it is compared to the respective lists to determine if it can be classified as a throw(), prep() or join(). If it can be, it is removed from the quote() and re-classified accordingly. A similar process if performed for the first word of each classified capital().

The next step of parsing process 100 is step 112, in Figure 2, wherein the classified words and/or phrases are manipulated to extract the most relevant terms therefrom. Step 112 is described with reference to Figures 4a through 4e, which illustrate sub-steps of step 112. Specifically, at step 250, a check is first performed to ensure that search data 28 contains groups (either a word or phrase) which has been classified as other than throw(). If all groups in search data 28 are classified as throw(), an error message is presented to the user instructing them to rewrite their search data at step 254. Otherwise, all groups in search data 28 which have been classified as throw()'s are discarded at step 258.

Next, a determination is made at step 262 as to whether the first remaining group in search data 28 is classified as phrase(). If the first remaining group is classified phrase(), then a determination is made at step 266 as to whether any group exists in search data 28 which has been classified as capital() or quote() and which is not immediately preceded with a group classified as prep() or join(). If one or more such groups are present in search data 28, the first such group's classification is changed at step 270 to rank1(). If, at step 266, it is determined that no such group exists in search data 28, the classification of the first group is changed from phrase() to rank1() at step 274.

If, at step 262, the first remaining group is not classified phrase() then a determination is made at step 278 as to whether the first remaining group in search data 28 is classified number(), adject(), or qword(). If the first remaining group is one of these classifications, a determination is made at step 282 as to whether any group exists in search data 28 which has been classified as capital() or quote() and which is not immediately preceded with a group classified as prep() or join(). If one or more such groups are present in search data 28, the first such group's classification is changed, at step 286, to rank1().

If, at step 282, no such group classified as capital() or quote() exists, a determination is made at step 290 as to whether there is any remaining group in search data 28 which is classified phrase(). If there is at least one such group, the classification of the first of these groups is changed to rank1() at step 294.

If, at step 290, there is no such group then a determination is made, at step 298, as to whether there is a remaining group in search data 28 which is classified number() or adject(). If there is at least one such group, the classification of the first of these groups is changed to rank1() at step 302.

If, at step 298, there is no such group, the first remaining group in search data 28, which was classified qword(), is changed at step 306 to a rank1() classification.

If, at step 278, it was determined that the first remaining group in search data 28 was not classified number(), adject() or qword(), then a determination is made at step 310 (in Figure 4b) as to whether the first remaining group is classified as capital() or quote(). If the first remaining group is classified as capital() or quote(), it is changed to a classification of rank1() at step 314.

If, at step 310, the first remaining group is not classified as capital() or quote(), then a determination is made at step 318 as to whether the first remaining group is classified as prep(). If it is, then at step 322 a determination is made as to whether any group exists in search data 28 which has been classified as capital() or quote() and which is not immediately preceded with a group classified as prep() or join(). If one or more such groups are present in search data 28, the first such group's classification is changed at step 326 to rank1().

If, at step 322, it is determined that no group classified as capital() or quote() exists in search data 28 that is not immediately preceded by a group classified prep() or join(), then at step 330 a determination is made as to whether any group exists in search data 28 which has been classified as phrase() which is not immediately preceded with a group classified as prep() or join(). If one or more such groups are present in search data 28, the first such group's classification is changed at step 334 to rank1().

If, at step 330, it is determined that no group classified as phrase() exists in search data 28 that is not immediately preceded by a group classified prep() or join(), then at step 338 a determination is made as to whether any group exists in search data 28 which has been classified as number() or adject() and which is not immediately preceded with a group classified as prep() or join(). If one or more such groups are present in search data 28, the first such group's classification is changed at step 342 to rank1().

If, at step 338, it is determined that no group classified as number() or adject() exists in search data 28 that is not immediately preceded by a group classified prep() or join(), then at step 346 a determination is made as to whether any group exists in search data 28 other than groups classified prep() or join(). If no such other groups remain in search data 28, then an error message is presented to the user at step 350. If such other groups do exist, then the first group of quote(), capital(), number(), adject(), or qword classification is changed to rank1() at step 354.

If, at step 318, the first group is not classified as prep(), then at step 358 (Figure 4c) a determination is made as to whether the first remaining group is classified as a join(). If it is, this group is deleted from search data 28 at step 362 and processing reverts to step 262.

At step 366 (Figure 4d), the first remaining group in search data 28 is selected for examination. At step 370, a determination is made as to whether the group is a phrase(), number(), capital() or quote() classification and whether it is immediately preceded by a group which is a join(). If these conditions are met by the group being examined and if the join() which precedes the group is in turn preceded by a group classified as rank1(), then at step 374 the classification of the group is changed to also be rank1(), i.e. - rank1(IBM) join(and) phrase(compiler) becomes rank1(IBM) join(and) rank1(compiler).

If the conditions of step 370 are not met by the group, at step 378 a determination is made as to whether the group is an adject() which is immediately preceded by a join() and, if so, if the group which immediately precedes that join() is classified as rank1(). If these conditions are met, then the first following group which was classified phrase(), number(), capital() or quote() is changed to rank1() at step 382.

If the conditions of step 378 are not met, at step 386 a determination is made as to whether the group is classified as a phrase(), number(), capital() or quote() and if it is immediately followed by a group classified as join() which is in turn immediately followed by a group which is classified as rank1(). If these conditions are met, then the classification of the group is changed to rank1() at step 390.

At step 394, a determination is made as to whether all of the groups in search data 28 have been considered. If not, the next group is selected for consideration at step 398 and processing returns to step 370. Once, at step 394, it is determined that all remaining groups have been considered, processing continues at step 402 (Figure 4e).

At step 402, apostrophe s's ('s) are deleted, if present, from each non-join() group. Next, at step 406, the first remaining group which is not classified as a join() is examined. A determination is made at step 410 as to whether the group which immediately precedes this group is a join(or). The term "join(or)" refers to the word 'or', from Table 3 of Appendix A, which will have been classified as a join(). If the condition at step 410 is true, then at step 414 any other join()'s which immediately precede the join(or) are removed.

Next, at step 418, a determination is made as to whether the immediately preceding non-join() is an or(). An or() is a classification for a list of search data words which can be separated by boolean OR's. For example, search data, "A or B or C" is re-expressed as or(A, B, C) for efficiency and convenience reasons. If, at step 418 the immediately preceding non-join() is an or(), then the preceding or(), the join(or) and the non-join() groups are combined.

If, at step 418, the preceding Non-join() is not an or(), processing continues at step 426 wherein the preceding non-join(), the join(or) and the non-join() group are combined.

As an example of steps 418, 422 and 426, given search data which has been classified as "phrase(pass) prep(from) capital(Tinkers) join(or) capital(Evert) join(or) capital(Chance)", when group "capital(Evert)" is processed at step 418, the processing will proceed to step 426. At step 426, the search data is combined to read, "phrase(pass) prep(from) or(capital(Tinkers), capital(Evert)) join(or) capital(Chance)". Next, when processing group "capital(Chance)" at step 418, the processing will proceed to step 422 wherein the search data is combined to read, "phrase(pass) prep(from) or(capital(Tinkers), capital(Evert), capital(Chance))".

At step 428 a determination is made as to whether all non-join() groups have been considered and, if not, the next group is selected at step 432 and processing reverts to step 406. If, at step 428, it is determined that all non-join() groups have been considered, the manipulation process is complete, as indicated at 436.

Once the manipulation of the classified words at step 112 is complete, step 116 of parsing process 100 is performed to complete the process. Specifically, in step 116 an examination is performed on each remaining group in search data 28 to determine groups which can advantageously be translated and/or enhanced. A translation table (not shown) of words and phrases and their preferred alternatives is maintained by process 100 and the remaining groups in search data 28 are compared to the entries in this table. For each match, the matching group is replaced with the preferred alternative, either explicitly or via a translation function.

For example, the translation table can contain an explicit entry for "get in touch" for which a preferred alternative can be "contact". Any group in search data 28 which contains the phrase "get in touch" will have this phrase replaced by "contact". As another example, the translation table can contain a function to convert time-related words into numeric equivalents. Specifically, any group in search data 28 containing the word "today" will have this word replaced with the current date in an appropriate format, such as dd/mm/yy. Similarly, whole numbers can be converted to text form, i.e. "7" converted to "seven".

Finally, step 116 can perform a synonym expansion for selected words and/or phrases. For example, the word "discover" can be expanded to "discover or invent or find".

Referring again to Figure 1, Natural Language Query Processor 24 passes the processed search data 28 to meta search engine 32. Meta search engine 32 receives processed search data 28 and further processes it to place it into forms suitable for the search engine or engines 36 which are defined for the information sources to be searched. For example, if the information sources to be searched are WWW pages, search engines 36 can be appropriate search engines such as Lycos, AltaVista, etc. Or, if a commercial database is to be searched, such as Lexis, search engines 36 can be the database's proprietary search engine. In any case, meta search engine 32 is responsible for assembling queries which are appropriate to each search engine 26 from processed search data 28.

In a present embodiment of meta search engine 32, queries are assembled for three search engines 36, specifically the AltaVista, Lycos and Excite search engines for WWW pages. As will be apparent, fewer or more search engines can be employed if desired. It is also contemplated that different sets of search engines can be employed for different subject matters. For example, general enquiries may be passed to the set of three search engines mentioned above, while an enquiry relating to legal issues may be sent to any two of these search engines and to the Lexis database. It is contemplated that the selection of an appropriate set of search engines can either be performed explicitly by the user, or implicitly by the search system 20, based upon recognized keywords in the processed search data 28 or other information such as the user's identity, location, etc.

As shown in Figures 5, 5a and 5b, at step 500 a set of search engines is selected. As mentioned above, this can be either an implicit selection (a default set) or an explicit selection made by the user or by the search system 20. Next, at step 504, search data 28 is examined and all groups classified as qword() are removed from the processed search data 28. Next, at step 512, a database of search engine capabilities, requirements and addresses (URL's or other appropriate address information) is consulted to determine the appropriate parameters for each search engine in the selected set of search engines.

If one or more boolean-type search engines such as Excite, AltaVista, etc. are included in the set of search engines, at step 516, search data 28 is simplified for such engines.

Figure 5a shows a simplification for such boolean engines wherein at step 550, the groups in search data 28 are sorted by classification, with the presently preferred sort order being rank1(), or(), capital(), quote(), phrase(), adject() and number(). At step 554, each or() group is changed to the syntax required by the search engine, for example or(phrase(a), capital(b), phrase(c)) can be converted to (a or b or c). At step 558, the first portion of the query for the boolean search engine is formed by combining all of the groups which were classified as rank1(), separated by AND's.

At step 562, a determination is made as to whether the next remaining group is classified as capital() or quote() and, if it is, that group is added to the query with an AND at step 566. If, at step 562, the next group is not a capital() or quote(), at step 570 multiple word phrases are split into individual words and combined with OR's and the resulting structure is add to the query with an AND. Next, at step 574, all or()'s are added to the query with an AND and, at step 578, all remaining unique words in the search data are combined into a structure, wherein each word is separated by an OR, and the resulting structure is added to the query with an AND.

If one or more "word-only" type search engines such as Lycos, HotBot, etc. are included in the set of search engines, at step 520, search data 28 is simplified for such engines. Specifically, as shown in Figure 5b, at step 600 the groups in search data 28 are sorted by classification, with the presently preferred sort order being rank1(), or(), capital(), quote(), phrase(), adject() and number(). Next, at step 604, the contents of all of the groups are examined to remove duplicate words in a group, or between groups.

At step 608, a number n is determined as being the number of words remaining in search data 28, if less than four, or the value $\log_3$ (number of words - 3). Next, at step 612, a determination is made as to whether the selected search engine accepts an input representing the "number of words to be matched" to have a 'hit'. If the engine does support this input, as determined from the information in database 512, then at step 616 the query is composed and consists of all of the words and n. If, at step 612, the engine does not support a "number of words to be matched" input then at step 620 the query is composed and comprises the first n words.

If one or more other search engines, such as Lexis, etc. are included in the set of search engines, at step 524, search data 28 is appropriately simplified for such engines as will be apparent, to those of skill in the art, in view of the above.

Referring again to Figure 1, the simplified queries 38 from meta-search engine 32 are dispatched to the corresponding search engines 36 via suitable transmission means. For example, if a search engine is accessible from a web page on the internet, the query is sent to the URL for that web page with the query being in the required format. As will be apparent to those of skill in the art, the present invention is not limited to internet and/or World Wide Web-based search engines and any accessible search engine can be employed.

Examples of such search engines include, but are not limited to, those accessible via a LAN, a dedicated telecommunications line, a dial-up telecommunications link, etc., or even one or more search engines integral with system 20 can all be employed with the present system.

At step 532 in Figure 5, 'hits' 42 (in Figure 1) from each search engine are received by meta-search engine 32. These hits are then passed to Search Results Filter 46 when results have been obtained from all of the search engines in the set or when a predetermined time limit has been exceeded without receiving results from one or more search engine.

The hits received by Search Results Filter 46 are generally in the form of an address, such as a URL, at which a relevant information source can be located and the identity of the search engine which returned the hit. Search Results Filter 46 combines the hits from each search engine into a single list and removes redundancies. The culled list of hits is placed into the format necessary to retrieve the individual information sources and this formatted list is transferred to Information Retrieval means 50.

From this formatted list, Information Retrieval means 50 retrieves the complete information sources 54 for each of a preselected maximum number of hits from each search engine 36. For example, the first 10 hits from each engine, after redundancies have been removed, may be retrieved.

The retrieved information sources are then examined by the Selector means 58. Selector means 58 performs several functions, including ranking the relevancy of the information sources retrieved and identifying their relevant portions for output to the user.

The process for ranking of the information sources employs the processed search data 28 from Natural Language Query Processor 24. Specifically, as illustrated at step 680 of Figure 6, a scoring regime is established for the retrieved information sources relative to the processed search data 28 and a score table is created to hold determined scores for each information source. A presently preferred scoring regime is given in Appendix B. In this regime, each group in processed search data 28 is treated as a separate candidate and separate totals are maintained for each candidate in the score table. An example of processed search data 28 which reads, "or(phrase(contact), phrase(personnel), phrase(names)); phrase(people); rank1(Gravis);

rank1(Logitech))" has four candidates.

At step 684, the processed search data 28 is augmented by adding the following to processed search data 28: for each group with multiple word phrases, create another group wherein the first word is capitalized, (i.e. - for phrase(big sky) create group phrase(Big sky)); for each group with multiple word phrases, create another group wherein each word is capitalized, (i.e. - for phrase(big sky) create group phrase(Big Sky)); for each group with multiple words, including any capitalized groups created in the preceding steps, another group is created by replacing spaces in the group with +'s (i.e. - for phrase(Mickey Mouse), create phrase(Mickey+Mouse)); and for each word, whether in a single word group or a multi-word group, make new words by capitalizing them. For example, the phrase(mickey mouse pluto) becomes phrase(Mickey), phrase (Mouse) and phrase(Pluto). Each of these created groups is then added to the score table, with a score for any of these groups being considered a score for the candidate, i.e. - a match with the augmented phrase(Mickey mouse) is scored for the phrase(mickey mouse).

Next, at step 688, a first retrieved information source is selected. At step 692, the information source is examined to determine each match between its contents and the groups in the score table. For each match, an entry is made in the score table for the corresponding candidate including the score assigned the match under the selected scoring regime and the location of the match within the information source.

Next, at step 696, the matches are sorted by their location within the information source. At step 700, a determination is made as to whether more than three matches were found within the information source. If three or fewer matches were found, the information source is assigned a rank of zero at step 704 and, if at step 706 it is determined that one or more information sources remain to be considered, the next information source is selected at step 708 and processing returns to step 692.

If at step 700 it is determined that more than three matches have been found in the information source, processing proceeds to step 712, shown in Figure 6a, wherein the first three consecutive matches are selected for further consideration. At step 716, a table is established with an initial score value for each candidate. An example of a table of presently preferred initialization values is given in Appendix C.

At step 720, the scores are determined for the set of three hits, referred to herein as a segment. Specifically, these segment scores are determined by adding the scores of the corresponding candidates in each match with the initial score value for each respective candidate, from Appendix C, to obtain total scores for each candidate for the segment. These candidate totals are then multiplied together, including candidates which were not represented in the segment and which thus only have their initial value. This value is then divided by the length of the segment (i.e. the number of characters, including white space, etc. between the start of the first match being considered and end of the last match being considered).

The result of this calculation is then multiplied by the value $\log_{10} (x)^{1.5}$, where x is the previously determined length of the segment. This latter step weights the result against segments which are relatively small. Finally, the result of this calculation is divided by the value $1+\log_{10} (y)$, where y is the difference between the number of matches in the candidate with the greatest number of matches and the average number of matches for the other candidates, however if the value of y is determined to be less than one, it is set at one. This calculation is intended to weight the result against segments with a high number of matches in just a few candidates and few matches in the remaining candidates. The result of all of these calculations is the resultant segment score.

A step 724, a determination is made as to whether all matches in an information source have been considered. If unconsidered matches exist, the next three consecutive matches are selected for consideration as a segment at step 728. In the event that less than three unconsidered matches exist, a segment of three is formed at step 728 by "padding", namely by taking the last three consecutive matches, even if one or two of these matches have previously been considered. Processing then commences again at step 716.

If, at step 724, it is determined that all matches have been considered, the two segments with highest scores are selected at step 732. It will be apparent to those of skill in the art that, in the event that only a single segment exists in an information source, processing will proceed from step 724 to step 764, described below.

At step 740, as shown in Figure 6b, the first of the two highest scoring segments is selected. At step 744, the selected segment is augmented by adding the immediately preceding match (if any) to form an augmented segment. As referred to herein, a segment is merely a first offset from the start of the information source defining the start location of the portion of the information under consideration and a second offset defining the end of the portion of interest in the information source. Thus, in step 744, the augmentation is accomplished by moving the first offset appropriately, towards the start of the information source. Similarly, when a segment is "scanned" or otherwise processed, the information source is actually being considered, between the two offsets.

Steps 716 and 720 are then performed again on this augmented segment. At step 748, the selected segment is augmented by adding the immediately following match (if any) to form a second augmented segment and steps 716 and 720 are then performed again.

At step 752, a determination is made as to whether the resulting score of either of these augmented segments is higher than the previous score for the segment. If at least one score is higher, the augmented segment with the highest score is selected at step 756 and steps 744 through 752 are performed again on the selected augmented segment, wherein the selected augmented segment is augmented to form two new augmented segments which are scored and compared to the score of the selected augmented segment.

This process of augmenting, scoring and comparing continues until it is determined, at step 752, that neither of the augmented segments have a score higher than the score of the previously selected segment. Once this is determined, the previous selected segment is deemed to be the result for the segment at step 760. A determination is made a step 762 as to whether the second highest scoring segment from step 732 has been considered and, if not, processing proceeds from step 744 for that segment. If both segments have been considered, then at step 764 the segment, whether augmented or not, with the highest score is deemed to be the segment of interest for the information source.

A determination is made at step 768 as to whether any other information sources remain for which a segment of interest has not been determined and, if this is the case, processing reverts to step 708. Otherwise, processing proceeds to step 800, as shown in Figure 6c.

At step 800 the final segment from each information source is ranked in descending order, by their respective determined scores. At this point, it is likely that these segments define portions of their respective retrieved information sources which are incomplete to some extent, such as only being portions of paragraphs and/or sentences. Further, if the information sources were World Wide Web pages, in HTML format, it is possible that one or more HTML tags are missing from the portions, rendering them unparsable by an HTML browser. Accordingly, at step 804, the final segments are "cleaned up". As this clean up process proceeds, the information source retrieved is modified, if necessary, by moving, adding or deleting information therein.

Specifically, if the retrieved information sources are HTML formatted files, then the retrieved information source is scanned, as indicated at step 900 in Figure 7, to determine if a <BODY> tag is present within the portion of the retrieved information source which is between the start and end points defined by the segment. If no such tag is present, then at step 904, the retrieved information source is scanned, commencing at the start defined by the segment and working towards the beginning of the retrieved information source, for HTML tags. For each tag encountered, the actions listed in the table in Appendix D are performed accordingly. For example, if a </CODE> tag is encountered, the tag is moved to the start of the segment and the scan is continued. As another example, if a <DD> tag is encountered, the tag is moved to the start of the segment and the scan is stopped. As another example, if a <TITLE> tag is encountered, the tag is not moved to the start of the segment and the scan stops. In the absence of a tag which stops the scan, the scan terminates when the beginning of the retrieved information source is encountered.

If, at step 908, a <BODY> tag is present, the segment start is updated to exclude the tag and all material before it.

Next, a determination is made at step 912 as to whether the segment includes a </BODY> tag. If no such tag is present, then at step 916, the retrieved information source is scanned, commencing at the end defined by the segment and working towards the end of the retrieved information source, for HTML tags. For each tag encountered, the actions listed in the table in Appendix E are performed accordingly. For example, if a </CAPTION> tag is encountered, the tag is moved to the end of the segment and the scan is continued. As another example, if a </LI> tag is encountered, the tag is moved to the end of the segment and the scan is stopped. As another example, if an <ADDRESS> tag is encountered, the tag is not moved to the end of the segment and the scan stops. In the absence of a tag which stops the scan, the scan terminates when the end of the retrieved information source is encountered.

If, at step 912, it is determined that the segment does include a </BODY> tag, the segment end is updated to exclude the tag and all of the material following it.

Next, at step 924, the "cleaned up" segment is scanned again, from the updated start to the updated end, to close any "open" tags (i.e. - an open tag for which there is no corresponding closing tag, e.g. <CAPTION> without a </CAPTION>) by adding the corresponding closing tag and to open any "dangling" tags (i.e. - closing tags without a corresponding open tag) by adding the corresponding open tag. As will be apparent to those of skill in the art, added closing tags will be added to the end of the segment, in reverse order to the order the corresponding open tags are encountered in the segment and added open tags are added to the beginning of the segment, in reverse order to the order the corresponding closing tags are encountered in the segment.

Next, at step 928, problematic tags are modified or removed in accordance with the table in Appendix F. Specifically, the segment is checked for any filenames present with tags, such as <a> or <IMG> tags, which are expressed in with relative names, i.e. - not with full universal resource locators (URL's). Any such filenames are converted to absolute names, with full URL's. Tags listed in (2) of Appendix F are removed from the information source, along with their contents, and the segment start and end are updated appropriately. Tags listed in (3) of Appendix F are removed from the information source, leaving their contents. Finally, the specific tags listed in (4) of Appendix F are altered in the indicated manner.

At step 932, each URL (hot link) within the segment is checked to confirm that it links to a valid/existing information source. If a URL does not link to a valid information source, the URL is "unlinked", but its text is left in place. If the URL does link to a valid information source, a check is performed to determine if one or more of the groups in processed search data 28 are present in the URL or in the information source to which it points. If one or more groups are present, this information source is retrieved by Information Retrieval means 50 and processed by Selector means 58. The final segment determined for this retrieved information source is ranked against the final segments previously determined for the other retrieved information sources and is added to the sorted final segments obtained at step 800. The clean up operations of step 804 are then performed on this latest, final segment.

The retrieval of information sources which are linked to previously retrieved information sources is limited to a preselected number of levels of recursion. It is contemplated that this number of levels of recursion will be a selectable parameter, although a suitable number of levels of recursion can be specified as a fixed default, if desired. In a present embodiment of the invention, no recursion (zero levels) is the selected default., but it is contemplated that more levels may be desired in other circumstances.

If, at step 804, the information source contains only text, i.e. - is not an HTML document, then the clean up proceeds as shown in Figure 8. Specifically, at step 950, the information source is scanned, from the start defined by the segment to the start of the information source, until the first blank line is encountered or the start of the information source is reached. If, as determined at step 954, a blank line was encountered, the segment start is updated at step 958 to include all material up to the blank line. If, as determined at step 954, the start of the information source is encountered, the segment start is updated at step 962 to include all material up to the beginning of the source.

Next, at step 966, the information source is scanned, from the end defined by the segment to the end of the information source, until the first blank line is encountered or the end of the information source is reached. If, as determined at step 970, a blank line was encountered, the segment end is updated at step 974 to include all material down to the blank line. If, as determined at step 966, the end of the information source is encountered, the segment end is updated at step 978 to include all material down to the end of the source.

As will be apparent to those of skill in the art, if information sources in formats other than text or HTML are retrieved, appropriate clean up operations will be performed, as desired.

As a final step of Selector means 58, the highest ranked, "cleaned up" segment is selected for output to the user, as is each cleaned up segment whose score is no less than a preselected level. In a present embodiment of the invention, up to the ten highest scoring segments whose scores are greater than 0.01 are output to the user as a first set and a second set of up to the next ten highest scoring segments whose scores are greater than 0.01 are also available for output to the user. As will be apparent to those of skill in the art, the selection of this output criteria is arbitrary and may be varied as desired but this criteria has been found to provide reasonable results.

Output device 62 then outputs the portions 66 of the cleaned up information sources indicated by the selected segments to the user. In a present embodiment of the invention, the output portions include a header which identifies the ranking of the portion, a link (URL) to the original information source (if appropriate), a number indicating the size of the original information source and a link (if appropriate) to the Search Engine 36 with which the information source was found.

An example of the operation of an embodiment of the present invention is given below. In the example, the user has entered "Where do Monarch butterflies spend the winter?" as the Natural Language Search Data 28. The processed search data from the Natural Language Query Processor 24 is "rank1(Monarch) phrase(butterflies spend) phrase (winter)" and this is passed to meta search engine 32.

In this example, a set of search engines 36 has been previously selected and includes the Lycos, AltaVista and Excite engines. Meta search engine 32 simplifies the processed search data 28 for each search engine in the set to obtain simplified search data appropriate to each engine. Specifically, for the Lycos engine, the search data which is dispatched is, "Monarch+butterflies+spend+winter". For the AltaVista search engine, the search data is, "Monarch+AND+(butterflies+OR+spend)+AND+(winter) Monarch ranked first". Finally, for the Excite search engine, the search data is, "Monarch+AND+(butterflies+OR+spend)+AND+(winter)". This search data is appropriately combined with the URL for each respective search engine and is transmitted to the search engine.

Again, in this example it has been previously decided that no more that the first twenty 'hits' from each search engine will be considered. Appendix G shows the actual HTML pages returned by each search engine and Appendix H shows the list of URL's which have been extracted from the pages in Appendix G, after obvious redundancies have been eliminated. In the Appendix, the URL's located by the AltaVista engine are identified with a "av##" prefix, those located by the Lycos engine are identified with a "ly##" prefix and those located by the Excite engine are identified with a "ex##" prefix. As will be noted, there was one redundant 'hit' in the first twenty URL's located by AltaVista, resulting in only nineteen entries for AltaVista in the list of Appendix G. Similarly, there were two redundant 'hits' in the first twenty URL's located by Lycos, resulting in only eighteen entires for Lycos in the list of Appendix G. In the cases wherein a redundancy is determined between the hits returned by two or more search engines, the highest ranked hit is retained and the other hit or hits are removed from the search engine results wherein they were lower scored. For example, if the Lycos search engine ranked a hit as being number two and Excite ranked the same hit as being number ten, and AltaVista ranked the same hit as being number seven, the Lycos hit is retained and the other two hits are removed from the hit lists.

Information retrieval means 50 then retrieves each of the information sources listed in Appendix G, if possible, and these retrieved information sources are processed by Selector means 58 to obtain the list of cleaned up final segments shown in Appendix I. This list includes the URL to retrieve the information source, the start and end points of the cleaned up segment (expressed as byte offsets from the beginning of the information source), and the score assigned to the information source by Selector means 58.

Appendix J shows the formatted text (converted from the raw HTML code) of two of the information sources retrieved from the information source listed in Appendix G and Appendix K shows the final segments from these information sources, as output to the user by output means 62.

As discussed above, the present invention allows a user to input a natural language query, search multiple and diverse databases, retrieve a plurality of information sources which are deemed relevant to the user's query and to extract the relevant portions of at least some of the information sources and present them to the user. It is contemplated that the present invention will assist the user by culling many information sources which are not relevant to the query and by extracting the relevant portions of the relevant information sources. Thus, the user will be presented with a concise selection of information which is relevant to the original query.

It is further contemplated that the present invention can be employed to locate information sources located on telecommunications networks, such as the internet or dial up connections, or on computer networks such as intranets, extranets, LANS, etc.

The above-described embodiments of the invention are intended to be examples of the present invention and alterations and modifications may be effected thereto, by those of skill in the art, without departing from the scope of the invention which is defined solely by the claims appended hereto.